Capítulo 5 – Princípios de Projeto (Resolução Completa)

1. Quais os benefícios do Ocultamento de Informação?

Três principais benefícios: (1) Desenvolvimento em paralelo – diferentes desenvolvedores podem trabalhar independentemente; (2) Flexibilidade a mudanças – detalhes internos podem mudar sem afetar o restante do sistema; (3) Facilidade de entendimento – cada parte entende apenas o necessário.

2. O que ocorre se várias classes forem movidas para o mesmo arquivo?

Melhora a coesão (por estarem próximas), mas prejudica o ocultamento de informação, pois aumenta o acoplamento entre classes.

3. O que é 'classitis' e por que é ruim?

É o excesso de classes pequenas, que aumenta o acoplamento e a complexidade geral do sistema, dificultando entendimento e manutenção.

4. Quais são os tipos de acoplamento?

(a) Aceitável – depende apenas da interface pública; (b) Ruim – depende de detalhes internos; (c) Estrutural – dependência explícita; (d) Evolutivo – dependência implícita entre mudanças.

5. Dê exemplos de acoplamento estrutural aceitável e ruim.

Aceitável: uso de classes da biblioteca padrão (ex: Hashtable). Ruim: acesso direto a atributos internos de outra classe.

6. Pode haver acoplamento sem referência explícita?

Sim. Por exemplo, duas classes que usam o mesmo arquivo ou variável global estão acopladas indiretamente (acoplamento evolutivo).

7. Qual problema há em colocar todo o código no método main?

Baixa coesão – o método acumula várias responsabilidades, violando o princípio da responsabilidade única.

8. Que princípio é violado no código onclick()?

Viola os princípios da Responsabilidade Única e de Demeter, pois mistura interface, lógica e persistência.

9. Se fosse preciso eliminar um conceito OO, qual seria?

A herança, pois pode ser substituída por composição. Encapsulamento e polimorfismo são mais fundamentais.

10. O que há de errado no método sendMail?

Viola o Princípio de Demeter (fala com o amigo do amigo). Deve mover a responsabilidade para ContaBancaria.

11. E no método imprimeDataContratacao()?

Também viola o Princípio de Demeter. Deve existir um método getDataContratacaoFormatada() em Funcionario.

12. O que está errado com as pré e pós-condições da subclasse?

Viola o Princípio de Substituição de Liskov, pois torna pré-condições mais restritas e pós-condições mais fracas.

13. Qual o CBO e o LCOM da classe A?

CBO(A) = 5 (B, C, D, E, F). LCOM = 1 (um par de métodos sem atributos em comum).

14. Qual classe é mais coesa?

Classe A é mais coesa (LCOM=0) que a classe B (LCOM=3).

15. Por que LCOM mede ausência e não presença de coesão?

Porque conta pares de métodos que não compartilham atributos, ou seja, mede a falta de coesão.

16. Todos os métodos devem contar no LCOM?

Não. Construtores e getters/setters devem ser ignorados, pois distorcem o resultado.

17. Complexidade ciclomática depende da linguagem?

Não. Mede o número de decisões lógicas (if, while, etc.), independente da linguagem.

18. Dê um exemplo com complexidade mínima.

void exibeMensagem() { System.out.println('Olá!'); } – Complexidade ciclomática = 1.

19. Compare versões monolítica e OO.

A versão monolítica mistura tudo em um bloco, dificultando manutenção e testes. A versão OO separa responsabilidades, reduz acoplamento e melhora a extensibilidade.